Fault Tolerant Clustering in Dense Wireless Sensor Networks *

Akshaye Dhawan, Neal Shukla, Cassandra Chapman and Stephan Hennion
Ursinus College
Department of Mathematics and Computer Science
Collegeville, PA 19426
{adhawan, neshukla, cachapman, sthennion}@ursinus.edu

Abstract

Wireless Sensor Networks are an important focus of research in distributed computing due to their many envisioned applications. Self-organizing these networks using clustering algorithms has been studied extensively in the literature as a means to conserve energy in sensor nodes. In clustering, the network organizes around a small set of cluster-heads which then gather data from their local cluster, aggregate this data and transmit it to the base station. In this paper we present a model for adding fault-tolerance to clustering algorithms in dense sensor networks. Since sensor nodes are often deployed in harsh environments, they are prone to failure. Cluster-head failure can leave a cluster disconnected from the base station until the network reorganizes again. We use the density of the network to improve the lifetime of the network by about 80% while allowing for recovery from the failure of a cluster head. This comes at the cost of a slightly increased number of clusters.

1 Introduction

Wireless Sensor Networks (WSNs) have attracted a lot of research interest due to their applicability in security, monitoring, disaster relief and environmental applications. WSNs consist of a number of low-cost sensors scattered in a geographical area of interest and connected by a wireless RF interface. Sensors gather information about the monitored area and send this information to an external node known as the base station. The radio on board these sensor nodes has a limited range and allows the node to transmit over short distances. In most deployment scenarios, it is extremely expensive for each node to communicate directly to the sink and hence, the model of communication is to transmit over short distances to other peers.

In order to keep their cost low, sensors are equipped with limited energy and computational resources. The energy supply is typically in the form of a battery and once the battery is exhausted, the sensor is considered to be dead. Sensor nodes are also limited in terms of memory and processing capabilities. Hence, harnessing the potential of these networks involves tackling a myriad of different issues from algorithms for network operation, programming models, architecture and hardware to more traditional networking issues. For a more detailed survey on the various computational research aspects of Wireless Sensor Networks, see the survey papers [2, 4, 12, 19, 18].

In this paper we focus on the problem of self-organizing these networks into clusters [1]. The use of clustering based protocols is important in WSNs since clustering allows the sensor nodes to reduce their energy consumption significantly. Nodes organize themselves into local clusters with each cluster having one node designated as the cluster-head. Nodes then communicate to their local cluster-head which in turn transmits data from this cluster to the base station. This results in considerable energy savings since the transmission distance to the cluster-head is very small when compared to the distance to the base station. However, clustering does impose a significant drain on the battery of the cluster-head. Hence, protocols like LEACH (Low-Energy Adaptive Clustering Hierarchy) [11] rotate the cluster-head responsibility randomly between nodes so as to ensure fairness. Additionally, clustering allows the cluster-head to aggregate data before transmission, effectively reducing the amount of data being transmitted to the base station.

Our work in this paper focuses on making such clus-

*This material is based upon work supported in part by the National Science Foundation under Grant No. DMS-1003972. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

tering algorithms fault-tolerant. Since most clustering algorithms operate in rounds, where a cluster-head is elected and then used for the duration of the round, the failure of a cluster-head results in the nodes in that cluster being disconnected from base station for the remainder of the duration of this round. Failure is of particular concern in the context of WSNs since they are often deployed in harsh terrains resulting in a high likelihood of node failure. Our work attempts to devise models for the design of fault-tolerant clustering by using general principles that can be applied to any clustering protocol. We introduce two basic models - *MaxBattery* Recovery and *Random* Recovery. We then modify LEACH to test these models in Section 5 and examine the performance of these recovery algorithms for improved failure recovery. Our approach is specifically designed for dense networks since it leverages this as part of the recovery process.

The remainder of this paper is organized as follows. In Section 2 we briefly cover related work on the problem of clustering and fault-tolerance in Wireless Sensor Networks. In Section 3 we discuss modifications made to LEACH in order to allow nodes to detect the failure of a cluster-head. Section 4 introduces our recovery scheme for dense networks. Section 5 studies the performance of our algorithms by comparing them to standard LEACH. Finally, we conclude in Section 6.

2 Related Work

A key application of wireless sensor networks is the collection of data for reporting in various scenarios. A common theme in current research has been to view these networks as large-scale distributed systems and study various problems with the objective of examining common design techniques that emerge. For example, in our prior work [5, 6] we have presented a model for designing distributed solutions to the coverage problems in these networks that can also be applied to a host of other problems.

In Wireless Sensor Networks, there are two types of data reporting scenarios: event-driven and on-demand [3]. Event-driven reporting occurs when one or more sensor nodes detect an event and report it to the sink. In on-demand reporting, the sink initiates a query and the nodes respond with data to this query.

Clustering has been seen as an important means of conserving energy while routing data in WSNs. Several clustering algorithms have been proposed in the literature including randomized clustering (of which LEACH [11] is an example), lowest cluster-ID clustering [9] and highest degree of connectivity [7] clustering. In practice, randomized clustering coupled with cluster-head rotation has been shown to be an effective approach to clustering [11]. See [1] for a survey on clustering algorithms for Wireless Sensor Networks.

Fault Tolerance has been extensively studied in the broader context of distributing computing [16], and also in the context of Wireless Sensor Networks. [13] examines the connection between classical fault tolerance techniques and sensor networks and provide two case studies. [10] examines fault tolerance in clustering, but only looks at heterogeneous sensor networks where clustering is performed by special high energy gateway nodes that are much more powerful than regular sensor nodes. [14] examines fault tolerant clustering by formulating clustering as the k-fold dominating set problem. They give a probabilistic algorithm for a unit disk graph network. The authors go on to present fast approximation algorithms for the special cases of graphs with low arboricity in [15]. In [17], the authors consider clustering in a sensor network to be deployed in landslide detection applications. However, they define a fault as a sensor that gives incorrect values as opposed to a failed sensor.

LEACH (Low-Energy Adaptive Clustering Hierarchy) [11] is a self-organizing, adaptive clustering protocol based on randomization. The randomization allows LEACH to distribute the overhead of being a cluster-head evenly amongst the nodes in the sensor network. Additionally, LEACH uses data fusion to compress the data being sent to the base station, thereby reducing the energy wasted in transmitting redundant data. In LEACH, sensors elect themselves to be cluster-heads with a certain probability. This phase in known as the Advertisement Phase. These cluster-heads then broadcast their status to the other nodes. Nodes determine which cluster they belong to by looking at the strength of the signals received from the various cluster-heads.

Once all nodes are organized into clusters, each cluster-head creates a Time Division Multiplexed (TDMA) schedule for its cluster. This allows the sensor nodes in the cluster to turn off their transceiver except at the time slot allocated to them, thereby saving considerable energy. A round is completed when every node has gone through its transmission. At this point, the cluster-head aggregates the data collected from the cluster and transmits it to the base station.

LEACH has emerged as the primary clustering protocol in WSNs and has been shown to be effective and energy-efficient. Hence, we chose to implement our extensions over LEACH.

3 Failure Detection

A key first step in devising a model for fault-tolerant clustering is that of detecting cluster-head failure. In this step, the nodes in a given cluster must be able to detect the failure of their cluster-head so that they can initiate the Failure Recovery phase outlined in Section 4.

Based on the existing setup of LEACH [11], a cluster-head only transmits data to its subordinate nodes during the advertisement phase (also known as the selection phase). During a round, once selection has been performed, the only nodes transmitting data are the subordinate nodes. As explained in Section 2, once the nodes organize themselves into clusters, each cluster-head creates a schedule for the nodes in its cluster. Therefore, in the current state of the LEACH protocol, a node in the cluster has no idea as to whether its cluster-head has failed, since it will never hear from the cluster-head again. Thus, the failure of a cluster head effectively disconnects the entire cluster for the remainder of the round. Also, all transmissions made by nodes post the cluster-heads failure are lost since they never get sent to the base-station.

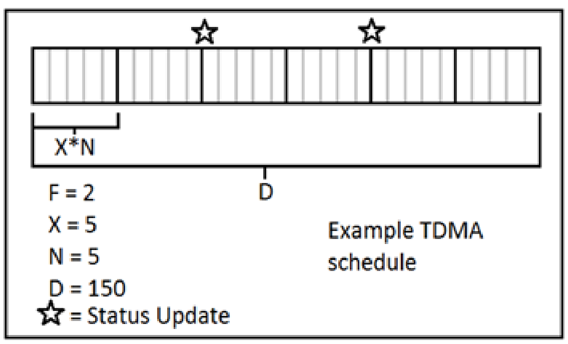

Figure 1. Failure Detection messages added to the TDM schedule

In order to implement detection of failure, we modify LEACH to add some periodic communication from the cluster-head to the nodes in the cluster. Consider the following situation - let a round begin at time t_1. Let the next round begin at time $t_1 + D$ where, D is the duration of a single round. Note that so far, this representation is exactly how a round works in LEACH. Now, suppose that for the cluster in question, there are N nodes that are a part of this cluster, each of which transmits for X seconds in the schedule for this cluster. Then, for this cluster there are $s = \frac{\frac{D}{X}}{N}$, sets of transmissions between the nodes and the cluster-head. We implement detection by having the cluster-head pick a random number F between 1 and s, where s is as defined above. After every F set of transmissions (i.e. at time $F * X * N$), all nodes turn their radio receiver back on and the cluster-head sends a small ping message indicating that it is up and running. If such a transmission is not received, the nodes can assume that the cluster-head has failed and employ the recovery algorithm.

Figure 1 illustrates the status update messages into an existing schedule for the given F, X, N and D values.

4 Failure Recovery

In this section, we present the design for our approach to cluster head failure recovery. Our approach is based on domination set theory. Briefly, a dominating set for a graph $G = (V, E)$ is a subset D of V such that every vertex not in D is connected to at least one member of D by some edge. More simply, given a graph G, a dominating set is a subset D of the vertices such that every vertex $v \in V$ is either in D or adjacent to a node in D. A k dominating set is one in which each node is adjacent to k nodes in the dominating set. A cluster head can be looked at as the dominating node for all the dominatees i.e., the sensors under it. Hence clustering is nicely modeled in this manner.

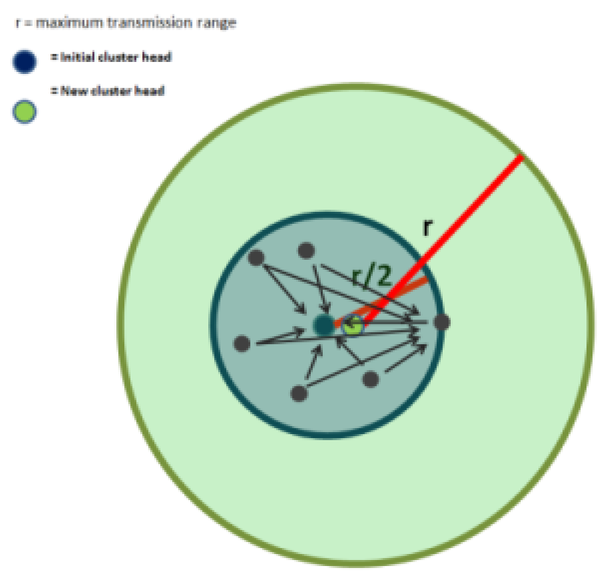

Figure 2. Figure illustrating cluster head recovery in a dense network

In order to recover from cluster head failure, the simplest approach to take would be to have a failed cluster head replaced by another node in the same cluster (i.e., to construct a 2-dominating set). In our work, we look at being able to deal with the failure of at most one cluster head per cluster. This essentially means that every cluster head must have one back-up node in the cluster. However, it is not straightforward to pick *any* node in the cluster as the replacement since this may leave some nodes in the cluster unreachable.

To allow any node in the cluster to serve as the back up, we restrict our initial clustering to $r/2$ where r is the maximum transmission range of any sensor node (assuming uniform nodes). This restriction is only possible in dense networks and we exploit the density to our advantage. Additionally, by reducing the range we also reduce the energy consumption of the sensor nodes. Most modern sensors have the ability to adjust their transmissions [5]. This allows us to select any node in the cluster to serve as a backup node to the cluster head in the eventuality of failure due to the following lemma:

Lemma: Given a sensor s_i transmitting with a range $r/2$, any sensor s_j that is centered at a point within the coverage range of sensor s_i, can transmit at a range r and cover all sensors previously covered by s_i.

The proof is trivial so we omit it for brevity. Figure 2 illustrates this point.

The lemma shows that if the initial clustering is at $r/2$, then the back up cluster transmitting at r can cover all the nodes previously covered by the failed cluster head. We experiment with two different approaches to selecting the back up node. In the first method called MaxBattery Recovery, we pick the highest energy node in the cluster to serve as the back up in the event of failure. This a purely greedy choice. We also implement Random Recovery wherein, a random node in the cluster is picked to serve as the backup cluster head. Both these approaches are studied through simulations in the next section.

5 Simulations

In order to study the performance of the proposed fault-tolerance model, we have conducted some preliminary simulations. We implement LEACH as described in [11] and then modify this implementation to start the initial clustering with half the maximum transmission range. We further implement both the MaxBattery and the Random Recovery models.

For our simulation setup, we randomly create dense networks of sensors. In a $50x50$m area, we place a varying number of sensor nodes with a maximum transmission range of 10 m. We also conducted simulations on $100x100$m and $200x200$m areas but these results are omitted for brevity. The general trends observed here applied to those simulations also.

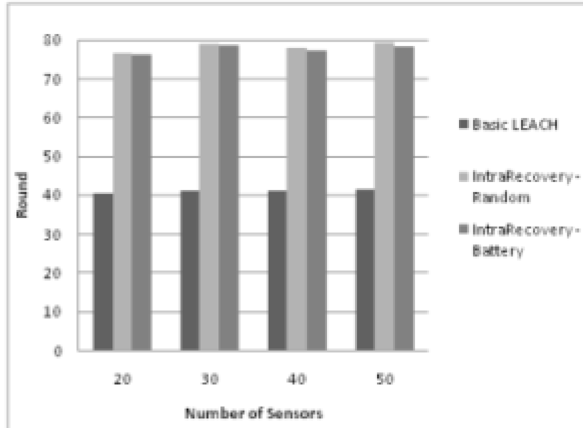

Figure 3. Average number of rounds all sensors are dead on

For the purpose of this simulation, we considered homogeneous nodes with identical batteries. The simulations also employ a linear power model where the energy required to transmit over a given distance is a linear function of the distance. In every round, 5% of the nodes volunteer to be cluster-heads. Node failure is injected randomly into the network upto a maximum of 5% of the cluster heads failing. The energy costs are modeled along those of real sensors with idle energy costs being comparable to receiving costs and transmission costs being approximately double that of reception [8].

Figure 3 presents the lifetime of the network. In this figure we track the number of rounds by which all the sensors have failed. As can been seen from the figure, the lifetime of Random Recovery is approximately 1.90 times as long as that of basic LEACH while the lifetime of MaxBattery Recovery is approximately 1.89 times as long. Because our recovery methods use a transmission distance of $r/2$ instead of r, they use significantly less battery than the basic LEACH algorithm. However, this is only feasible in dense networks.

Next, we present in Table 1 the average number of cluster heads for LEACH, Random Recovery and MaxBattery Recovery across 30 randomly generated networks for each data point. As can be seen from the table, as the number of sensors increases, the number of clusters for Random and MaxBattery Recovery also increases. This is due to the fact that each cluster head transmitting at half its maximum range can cover fewer sensors. However, this number is not significantly

Algorithm	n=20	n=30	n=40	n=50
Basic LEACH	2.13	2.93	4.06	4.88
Random Recovery	2.1	2.94	4.86	5.79
MaxBattery Recovery	2.13	2.93	4.88	5.14

Table 1. Average number of cluster heads per round in a 50x50m area

Algorithm	50x50	100x100	200x200
Basic LEACH	4.88	5.14	5.95
Random Recovery	5.79	5.75	6.45
MaxBattery Recovery	5.14	5.95	6.35

Table 2. Average number of cluster heads for a network of 50 sensors

higher and still does not impact our improvements in lifetime.

We also present a snapshot of this data for a network of 50 sensors in varying area sizes in Table 2. As can been seen, the general trend observed for a smaller area is preserved.

6 Conclusions

In this paper we presented an approach to designing fault tolerant clustering algorithms in dense sensor networks. Our approach shows significant improvement in lifetime and failure recovery while increasing the number of clusters slightly.

References

[1] A. A. Abbasi and M. Younis. A survey on clustering algorithms for wireless sensor networks. *Comput. Commun.*, 30:2826–2841, October 2007.

[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Commun. Mag.*, pages 102–114, 2002.

[3] J. Carle and D. Simplot-Ryl. Energy efficient area monitoring for sensor networks. *IEEE Computer*, 37(2), 2004.

[4] C.-Y. Chong and S. Kumar. Sensor networks: evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91(8):1247–1256, Aug. 2003.

[5] A. Dhawan and S. K. Prasad. A distributed algorithmic framework for coverage problems in wireless sensor networks. In *IPDPS '08: Proceedings of the 22nd IEEE International Parallel and Distributed Processing Symposium.* IEEE, 2008.

[6] A. Dhawan and S. K. Prasad. A distributed algorithmic framework for coverage problems in wireless sensor networks. *International Journal of Parallel, Emergent and Distributed Systems(IJPEDS)*, 2009.

[7] J. B. Dugan and K. S. Trivedi. Coverage modeling for dependability analysis of fault-tolerant systems. *IEEE Trans. Comput.*, 38:775–787, June 1989.

[8] L. Feeney and M. Nilsson. Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1548–1557 vol.3, 2001.

[9] M. Gerla and J. Tzu-Chieh Tsai. Multicluster, mobile, multimedia radio network. *Wireless Networks*, 1:255–265, 1995. 10.1007/BF01200845.

[10] G. Gupta and M. Younis. Fault-tolerant clustering of wireless sensor networks. *IEEE Wireless Communications and Networking, 2003*, pages 1579–1584, 2003.

[11] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *HICSS '00: Proceedings of the 33rd Hawaii International Conference on System Sciences-Volume 8*, page 8020, Washington, DC, USA, 2000. IEEE Computer Society.

[12] S. S. Iyengar and R. Brooks. Computing and communications in distributed sensor networks. *Special Issue, Jr. of Parallel and Distributed Computing*, 64(7), 2004.

[13] S. S. Iyengar and R. Brooks. Computing and communications in distributed sensor networks, 2004.

[14] F. Kuhn. Fault-tolerant clustering in ad hoc and sensor networks. In *In 26th International Conference on Distributed Computing Systems (ICDCS*, 2006.

[15] C. Lenzen and R. Wattenhofer. Minimum dominating set approximation in graphs of bounded arboricity. In *Distributed Computing*, volume 6343 of *Lecture Notes in Computer Science*, pages 510–524. 2010.

[16] D. K. Pradhan, editor. *Fault-tolerant computer system design.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.

[17] R. Raj and M. Ramesh. Fault tolerant clustering approaches in wireless sensor network for landslide area monitoring. In *Proc. of the 2008 International Conference on Wireless Networks (ICWN'08)*, pages 107–113, 2008.

[18] K. Römer and F. Mattern. The design space of wireless sensor networks. *IEEE Wireless Communications*, 11(6):54–61, December 2004.

[19] S. Schmid and R. Wattenhofer. Algorithmic models for sensor networks. In *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*, pages 11 pp.–, April 2006.